Departamento de Informática

Licenciatura em Informática

Computação Gráfica

4º Ano

Revisão da Linguagem de programação C

- Estruturas condicionais
- Entrada e saída de dados
- Laços
- Vectores
- Matrizes
- Funções
- Biblioteca Grafica C

Estruturas Condicionais

A instrução *if*

Formato 1:

```
if (expressao)
  instrução1;
```

Formato 2:

```
if (expressao)
  instrução1;
else
  instrução2;
```

Formato 3:

```
if (expressao)
  instrução1;
else
if (expressao)
            instrução2;
        else
            instrução3;
```

Estruturas Condicionais

- A instrução s*witch()*:

```
#include <stdio.h>
main()
{
   int resposta;

printf("Digite um numero entre 1 e 5: ");
   scanf("%d", &resposta);
```

Estruturas Condicionais

```
switch (resposta)
{
   case 1 :
   {
   prntf("Voce digitou 1.");
      break;
   }
   case 2 :
   {
      printf("Voce digitou 2.");
      break;
   }
   default : printf("Valor inválido. Tente
novamente!");
}
}
```

Funções básicas: Entrada/Saída

- *A função `printf()` (strings + variáveis):*
  - `printf("Era exatamente isto que eu gostaria de escrever!");`
  - `printf("\nO valor de x e %d", x);`
  - `printf("\n %c %d %f %s %u", car, int_sin, ponto_flut, palavra, int_não_sin);`
  - `printf("\n \a \b \t");`
- *A função `puts()` (somente strings):*
  - `puts("Hello, world.");`

Funções básicas: Entrada/Saída

- *A função* `scanf()`:
  - `scanf("%d", &x);`
  - `scanf("%f", &taxa);`
  - `scanf("%d %f", &x, &taxa);`

Laços de Repetição

- for():

```
for (contagem = 0; contagem < 100; contagem++)
   printf("\n%d",contagem);

for (contagem = 100; contagem > 0; contagem--)
   printf("\n%d",contagem);

for (contagem = 0; contagem < 100; contagem += 5)
{
   printf("%d",contagem);
   puts();
}
```

Laços de Repetição

- while():

```
#include <stdio.h>
int contagem;
main()
{
   /* Imprime os numeros de 1 a 20 */
   contagem = 1;
   while(contagem <= 20)
   {
      printf("\n%d", contagem);
      contagem++;
   }
}
```

Laços de Repetição

- do..while():

```
#include <stdio.h>
int contagem;
main()
{
   /* Imprime os numeros de 1 a 20 */
   contagem = 1;
   do
   {
      printf("\n%d", contagem);
      contagem++;
   } while(contagem <= 20);
}
```

Vetores

```
#include <stdio.h>
int vet[10], i;
main()
{
   for(i=0;i<10;i++) // Lê todo o vetor
      scanf("\n%d",&vet[i]);
   for(i=0;i<10;i++) // Mostra todo o vetor
      printf("\n%d",vet[i]);
}
// printf("%d",vet[vet[2]]);
// printf("%d",vet[2+3]);
```

Matrizes

Matrizes Bidimensionais:

```
#include <stdio.h>
#define MAX 2
int mat[MAX][MAX], i, j;
main()
{
   for(i=0;i<MAX;i++)     // Controle das linhas
       for(j=0;j<MAX;j++) // Controle das colunas
       {
           printf("Entre com o elemento
   [%d][%d]:",i,j);
           scanf("\n%d",&mat[i][j]);
       }
}
```

Funções

- Mecanismo básico

```
#include <stdio.h>

void ola_mundo(void);

main()
{
   ola_mundo();
}

void ola_mundo(void)
{
   puts("Olah mundo!");
}
```

```
#include <stdio.h>

tipo_retorno nome_funcao(argumentos);

main()
{
      chamada_da_funcao();
}

tipo_retorno nome_funcao(argumentos)
{
      instruções;
}
```

Funções – Escopo de variáveis

```
#include <stdio.h>
int x = 1, y = 2;
void demo(void)
{
   int x = 88, y = 99; // Declara e utiliza duas variáveis locais
   printf("\nDentro de demo(), x = %d e y = %d.", x, y);
} // Implementação sem protótipo!
main()
{
   printf("\nAntes de chamar demo(), x = %d e y = %d.", x, y);
   demo();
   printf("\nDepois de chamar demo(), x = %d e y = %d.", x, y);
}
```

Biblioteca Gráfica C

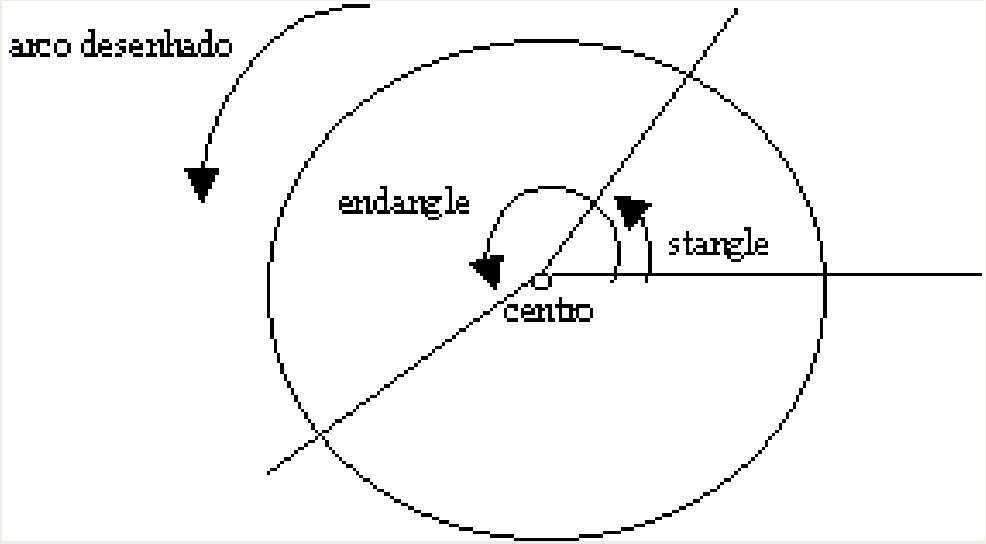
arc	bar	bar3d
circle	cleardevice	clearviewport
closegraph	detectgraph	drawpoly
ellipse	fillellipse	fillpoly
floodfill	getarccoords	getaspectratio
getbkcolor	getcolor	getdefaultpalette
getdrivername	getfillpattern	getfillsettings
getgraphmode	getimage	getlinesettings
getmaxcolor	getmaxmode	getmaxx
getmaxy	getmodename	getmoderange
getpalette	getpalettesize	getpixel
gettextsettings	getviewsettings	getx
gety	graphdefaults	grapherrormsg
_graphfreemem	_graphgetmem	graphresult

Biblioteca Graphics C

imagesize	initgraph	installuserdriver
installuserfont	line	linerel
lineto	moverel	moveto
outtext	outtextxy	pieslice
putimage	putpixel	rectangle
registerbgidriver	registerbgifont	restorecrtmode
sector	setactivepage	setallpalette
setaspectratio	setbkcolor	setcolor
setfillpattern	setfillstyle	setgraphbufsize
setgraphmode	setlinestyle	setpalette
setrgbpalette	settextjustify	settextstyle
setusercharsize	setviewport	setvisualpage
setwritemode	textheight	textwidth

void far arc(int x, int y, int stangle, int endangle, int raio)

- Desenha um arco de circunferência, com coordenada de centro definida por *x* e *y*. O arco começa no ângulo *stangle,* e termina em *endangle,* ambos em graus.
- O tamanho do raio é determinado por raio.



void far bar(int left, int top, int right, int bottom).

- Desenha uma barra (retângulo com preenchimento interno).
- A coordenada do canto superior esquerdo é determinada por *left* e *top*, e a do canto inferior direito por *right* e *bottom.* Usa a máscara de preenchimento definida por *setfillstile()*
- (caso nenhuma máscara tenha sido definida, como padrão é usado o preenchimento uniforme, com a cor de desenho atual).

Padrões de fundo

Legenda (da esquerda para direita, de cima para baixo)

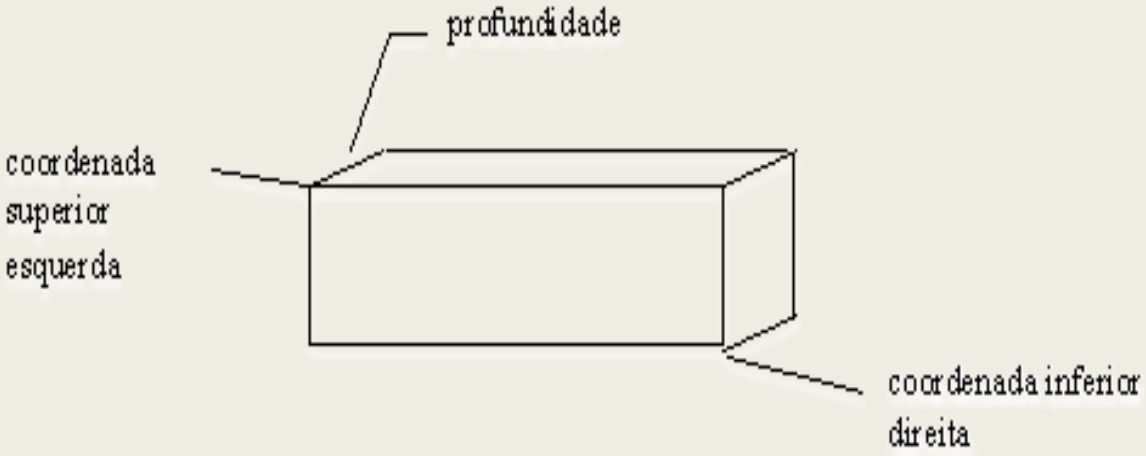
LINE_FILL	LTSLASH_FILL	SLASH_FILL	BKSLASH_FILL
LTBKSLASH_FILL	HATCH_FILL	XHATCH_FILL	INTERLEAVE_FILL
WIDE_DOT_FILL	CLOSE_DOT_FILL	EMPTY FILL	SOLID_FILL*

void far bar3d(int left, int top, int right, int bottom, int depth, int topflag)

Desenha uma barra com efeito 3D.

Depth indica a profundidade, em pixels, da barra. Se definirmos *topflag* igual a zero, a linha superior da barra não é desenhada.

Para qualquer valor diferente de zero, a barra é desenhada integralmente.



Mais funções

- **void far circle(int x, int y, int radius):** Desenha um círculo de raio *radius,* com centro nas coordenadas *x* (coluna) e *y* (linha).
- **void far cleardevice(void):** Limpa a tela gráfica.
- **void far clearviewport(void):** Limpa a janela de visualização corrente.

  Função relacionada: setviewport()
- **void far closegraph(void):** A função *closegraph()* fecha o sistema gráfico.

Mais Funções ..

- **void far floodfill(int x, int y, int border):** Preenche uma região delimitada por uma linha. A coluna do ponto de referência é determinada por *x*, e a linha por *y.*
- **int far getbkcolor(void):** Retorna a cor de fundo corrente.
- **int far getcolor(void):** Retorna a cor de desenho corrente.
- **char *far getdrivername(void):** Retorna um ponteiro para o nome do driver gráfico corrente.
- Função relacionada: *detectgraph()*

Mais Funções ..

- **int far getgraphmode(void):**
- Retorna o modo gráfico corrente.
- **char * far getmodename(int mode_number);**
- Retorna um ponteiro para uma string com nome do modo gráfico especificado. O modo 2, por exemplo, corresponde a 640 x 480 VGA.

Mais Funções ..

- **int far getmaxx(void) :** Retorna a máxima coordenada x (número de colunas menos um).
- **int far getmaxy(void):** Retorna a máxima coordenada y (número de linhas menos um)

Mais Funções ..

- **int far getx(void) :** Retorna a posição horizontal (coordenada x, ou coluna) atual da "caneta" na tela
- **int far gety(void):** Retorna a posição vertical (coordenada y, linha) atual da "caneta" na tela
- **unsigned far getpixel(int x, int y):** Retorna a cor do pixel especificado por *x*, coluna e *y*, linha.
- **char *far grapherrormsg(int errorcode) :**Recebe como único parâmetro um número de código que representa o erro ocorrido (este código é o retorno da função graphresult), e retorna uma string, que pode ser impressa na tela para explicar o que causou o erro.

Mais Funções ..

- **void far initgraph(int far *graphdriver, int far *graphmode, char far *pathtodriver)**
- Esta função tenta inicializar o modo gráfico de acordo com os parâmetros passados.

  Graphdriver é um ponteiro distante para inteiro que contém um único número indicando qual o driver que deve ser inicializado. De qualquer maneira, é possível usar uma opção especial chamada DETECT que tentará autodetectar o driver do seu dispositivo gráfico.

Mais Funções ..

- Graphmode é um ponteiro distante para um inteiro que avisa qual a resolução e a profundidade de cores que serão usados no modo gráfico.

- **Pathtodriver** é uma string que indica o caminho do arquivo *.BGI (Borland Graphics Interface) que contém o driver que a função vai precisar para inicializar o modo gráfico.

Mais Funções ..

- **void line(int x1, int y1, int x2, int y2):** Desenha uma linha que vai das coordenadas da tela x1,y1 até x2,y2.
- **void far outtext(char *far textstring) :** imprime uma string na tela do modo gráfico da mesma maneira que na função puts().
- **void far outtextxy(int x, int y, char *far textstring)**
- Como na função anterior, mas com esta é possível escolher onde exatamente o texto será impresso, como ao usar a função gotoxy() seguida de puts() no modo texto.
- **void far putpixel(int x, int y, int color)**
- Coloca um único pixel na tela, nas coordenadas (x,y) e com a cor color.

Mais Funções ..

- **void far rectangle(int left, int top, int right, int bottom):** desenha um retângulo com as coordenadas passadas como parâmetros, e pode ser usada para, entre outras utilidades, desenhar janelas e molduras para seus programas.
- **void setbkcolor(int color) :**Designa a cor que será usada no fundo da tela ou janela definida por setviewport.
- **void setcolor(int color):** Escolhe a cor usada para escrever, ou desenhar na tela, com funções como rectangle, outtext, circle, entre outras.

Mais Funções ..

- **void far setgraphmode(int mode) :** Muda o modo gráfico para o parâmetro mode.
- **void far setlinestyle(int linestyle, unsigned upattern, int thickness)**
- Determina o estilo da linha, o seu padrão, e a largura do traço. Consulte a ajuda online do seu compilador para saber quais são os modos disponíveis.
- **void far setviewport(int left, int top, int right, int bottom, int clip):** Abre uma nova "janela" dentro da tela nas coordenadas passadas como parâmetros, como na função window() do modo texto.

Mais Funções ..

- **void far settextstyle (int font, int direction, int charsize)**
- Muda as características dos caracteres impressos por funções como outtext e outtextxy, os parâmetros determinam a fonte usada (existem algumas pre-definidas pela biblioteca), a direção de escrita – HORIZ_DIR da esquerda para a direita, e VERT_DIR de baixo para cima -, e o tamanho da fonte (um valor de 1 até 10).